Lens-based Focus+Context Visualization Techniques for Interactive Exploration of Web-based Reachability Maps

Marc Listemann
Institute of Data Science
German Aerospace Center (DLR)
Germany
marc.listemann@dlr.de

Matthias Trapp
Hasso Plattner Institute
Faculty of Digital Engineering
University of Potsdam, Germany
trapp@hpi.de

Jürgen Döllner
Hasso Plattner Institute
Faculty of Digital Engineering
University of Potsdam, Germany
doellner@hpi.de

ABSTRACT

Reachability maps are powerful means to help making location-based decisions, such as choosing convenient sites for subsidiaries or planning vacation trips. Existing visualization approaches, however, introduce drawbacks concerning an effective acquisition of relevant information and an efficient data processing. In this paper, we introduce the first approach known so far to apply focus+context techniques to web-based reachability maps. We therefore propose a real-time enabled combination of an isochrone-based and a network-based representation of travel times obtained from multi-modal routing analysis using interactive lenses. We furthermore present a GPU-accelerated image-based method to compute isochrones based on a travel time-attributed network on client-side and thus achieve reduced data transmission efforts while yielding isochrones of higher precision, compared to generalized geometry-based approaches.

Keywords
focus+context, geovisualization, reachability maps, mobility analytics, web-based real-time rendering

1 INTRODUCTION

Reachability analyses aim to inform about the potential of an object to reach all locations in a given area from a starting point. To this effect, the "Single-source Shortest-path (SSSP)" problem has to be solved, as it finds all shortest paths between a starting point V_s and every possible node V_i in a graph-based network. The result are computed travel times for every node, representing the potential of an object using a specific transport medium at V_s to reach the respective nodes. As a result, several conceivable applications arise from reachability analytics, ranging from leisure time activity planning to complex site analysis [32].

1.1 Problem Statement

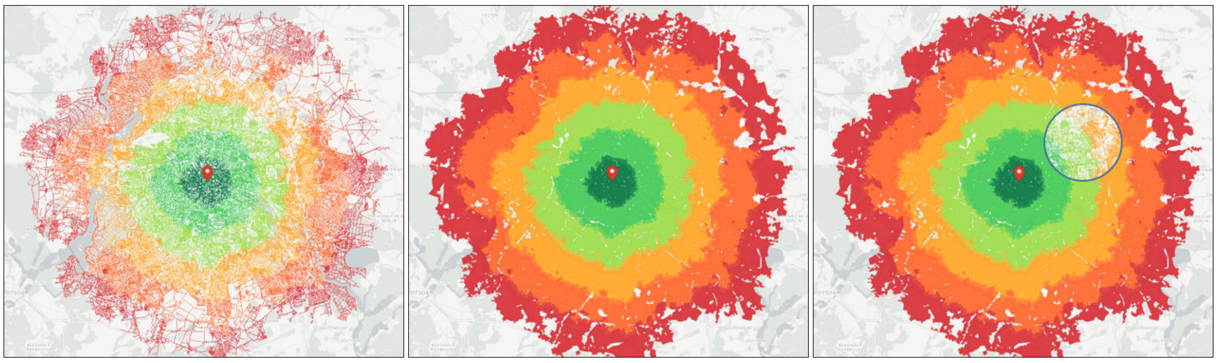
The computed travel times can consequently be visualized on a map where they are usually grouped into intervals by assigning them colors. Modern technologies allow for a web-based representation of reachability maps, thus providing a fast and interactive access to travel time information, whereas expensive computations are performed in the backend. With respect to 2D web-map applications, we emphasize two basic approaches for the visualization of reachability data: *Isochrone-based Visualization (IBV)* and *Network-based Visualization (NBV).*

In this paper, we refer to isochrones as generalized areas of equal travel times, being composed into intervals by assigning meaningful colors (Fig. 1b). The NBV, on the other hand, denotes a direct color mapping of travel time information onto the edges of the geographic network, as proposed by Schoedon *et al.* [23] (Fig. 1a). It therefore offers a high degree of detail, since the rendered lines directly refer to the streets in the network and no pre-generalization is applied, as it is the case for many isochrone computing procedures.

However, both approaches introduce implicit challenges regarding the user's potential to effectively obtain relevant information from the map:

C1: Isochronal representations lack a detailed view on the network due to generalization measures. Therefore, they may cause interpretation errors in areas of interest.

C2: Network-based representations impede a fast survey of the overall situation in the specified area concerning travel time information. They are likely to introduce interpretation difficulties and visual clutter at low zoom levels.



(a) Network-based visualization. (b) Isochrone-based visualization. (c) Lens-based visualization.

Figure 1: The basic concept of the lens-based focus+context visualization technique proposed in this paper. Two generated representations with different information density for focus and context, i.e., a network-based (a) and an isochrone-based representation (b) are combined to a single visualization using a magic lens metaphor (c).

C3: Conventional methods for generating isochrones rely on computation of geometry data and thus implicate a massive data transfer between server and client which involves latencies.

Hence, a concept is required that (1) combines the advantages of IBVs and NBVs (i.e., an overview-giving impression and a detailed view of reachability information) in a single visualization and (2) establishes a method for the generation of isochrones that addresses C3 and takes advantage of the properties of the Graphics Processing Unit (GPU), allowing for real-time rendering of massive data sets [30, 28]. The overall goal of this paper is therefore to support the user of a reachability map to effectively extract relevant information by retaining overview simultaneously while being provided with real-time enabled interaction methods.

1.2 Approach and Contributions

To this effect, *focus+context* is a powerful paradigm to improve the user's ability to extract relevant information out of a visualization as it represents detailed information in *focus* areas, which are embedded and linked to a *context* view, thus taking advantage of the available viewport space [6]. This work makes use of a prevalent Focus+Context Visualization (FCV) that is known as *Magic Lens* [27] and combines the two mentioned approaches to gain a user-defined focal view inside the surrounding isochrones (Fig. 1c), ensuring an appropriate and real-time visualization as well as a high degree of interaction. It is prototypical implemented as a web-map application using state-of-the-art web technologies, e.g., Web Graphics Library (WebGL) and HTML5 to obtain GPU-accelerated run-time performance.

For it, the GL Transmission Format (glTF) tiling approach by Schoedon *et al.* is being extended, while benefiting from GPU-enabled buffers containing the geographic network and the computed travel times [23]. Originally used for the rendering of travel time-attributed lines to obtain a detailed representation of reachability, it can be shown that this technique represents the fundamentals for the generation of an overview or context representation based on isochrones, too. For the isochrone generation, a GPU-based image-processing technique called Jump-Flooding Algorithm (JFA) [21] is adapted that allows for efficient client-side rendering. To summarize, this paper makes the following contributions:

1. It presents a concept and prototypical implementation of a lens-based FCV for reachability maps in the context of mobility analytics.
2. It describes an approach to generate isochrones using the JFA that is capable of real-time performance in a browser-based visualization environment.

The remainder of this paper is structured as follows. Sec. 2 reviews related work on interactive reachability maps and lens-based focus+context techniques. Sec. 3 introduces the concept of our approach and provides details for the proposed web-based isochrone generation method. Sec. 4 evaluates the results of that method, followed by a discussion of its run-time rendering performance as well as conceptual and technical limitations, and states possible future research directions. Finally, Sec. 5 concludes this paper.

2 RELATED WORK

The related work is surveyed with respect to previous approaches on travel time visualization and FCV techniques for 2D geographic data.

2.1 Reachability Maps

Previous work on reachability maps uses *IBV* [9, 3] or *NBV* [12, 23], whereas the latter does not scale for low zoom levels. In the work of Spiekermann, transportation networks are distorted into so-called *time-space maps* whose scale does not rely on spatial metrics but on travel time units [25]. Such representation is used by Lemke *et al.* to demonstrate shrinking regional disparities inside Europe in an economical context [18]. Whereas the time-space map approach may be supportive in this specific use case, it lacks obtaining concrete travel time values for a certain region. Besides that, an adaption to large scale applications such as web maps would not be sufficient as the map implicitly distorts outwards and small-scale distortions are hardly noticeable. Moreover, the authors indicate that the visualization may cause interpretation mistakes, as regions between the calibration points, which are considered major cities, could lose their accessibility, which is not represented in the map, in fact.

On the other hand, the work of Carden proposes an alternative displaying of reachability information within a map [5]. Similar to the tube map of London, a web-based application for the interactive reachability analysis was developed. Therefore, travel times from a selected subway station are computed and visualized by laying circular isochrones around the respective station and distorts the tube map in order to fit into the isochrones. A similar approach is used to visualize travel time data for cities in the Netherlands [19]. Instead of distorting a public transportation network, the map itself is distorted to fit the circular isochrones.

2.2 Focus+Context for Map Visualization

Focus+context research dates back to the 1980s, where Furnas developed "Generalized Fisheye Views" [8]. This distortion-based lens metaphors have been refined by various research [11]. Meanwhile, a number of different focus+context techniques have been established, though a generally accepted systematization is still missing [16, 6, 4]. However, Lokuge and Ishizaki applied these techniques to geographic data by developing an interactive map system that reacts on user queries [13]. It manipulates typography, transparency and color to alter the appearance of geographical elements due to the interest of the user. In this context, *Visugrams* enable simultaneous highlighting of relevant geographic information and related abstract information selected by the user [7].

Concerning focus+context applications for transportation networks, research has been examined for navigational problems. As such, Agrawala and Stolte present improved static route maps derived from generalization techniques [1]. They analyze hand-drawn route maps as well as cognitive aspects in psychology, to determine the characteristics of an effective route map, that hides dispensable streets and emphasizes significant route elements. A real-time enabled system for automatic generation of enhanced route maps is further being developed, performing sophisticated generalization steps. An improved method to this approach is furthermore introduced by Kopf *et al.* [15]. Karnick *et al.* , on the other hand, establish a multi-focal technique that generates detail lenses related to specific Point-of-Interests (POIs) along the route [14]. These should alleviate the readability of the map by magnifying those parts of the route that require an action by the driver. However, although their approach is carried out in a web-based automated system, it is developed for non-interactive, static maps too.

Instead of applying the frequently used fisheye distortion, Haunert and Sering suggest for a graph-based optimization approach in road networks to properly display a focal region while distorting only the sparse parts of the network [10]. Based on a user-defined Region-of-Interest (ROI) and zoom factor, the focus region is scaled up by that factor and the context is scaled down while minimizing the square sum of distortions. Though being able to run in polynomial time, the approach lacks real-time capability.

Krause *et al.* introduce a displacement technique for a public transportation system in Konstanz, Germany. They aim to visualize travel times between different network elements and to facilitate comparisons between different bus routes [17]. This is implemented by mapping computed travel times to visual distances of the edges. They develop i.a. a radial layout for representing the results of a routing query. It places bus stations concentric around a starting position and preserves topological characteristics by aligning the edges between the stations according to their geographic positions. However, this approach is can hardly be adapted to general transportation systems.

However, a real-time capable focus+context technique proposed by Wang and Chi, focus on the visualization of routes in public transport networks, e.g., metro maps [31]. By highlighting those parts of the subway network that are members of the best route to a destination and providing more space the related stations, they enable an adequate visualization of complex metro maps on small display areas. Additionally, to facilitate the map reading, they apply octilinear transportation lines and equal distances between the stations. The final layout is adapted due to the least square solving.

Furthermore, techniques such as semantic depth-of-field [16] can be applied to highlight geographic line elements on a map [29]. These allow for an assignment of relevance indicating values to the data and thereby carrying out the decision of an element being considered part of the focus or the context.

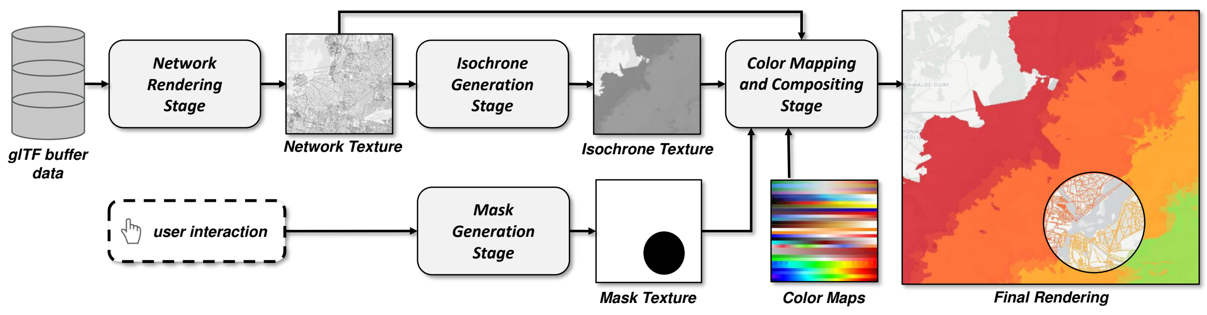


Figure 2: Conceptual overview of the presented approach to generate a FCV for reachability maps. Only the frontend is illustrated, as it is the focus of this work. Several rendering passes are executed to create three textures that are merged in a compositing pass.

3 VISUALIZATION APPROACH

This section presents the concept of implementing a FCV by combining IBV and NBV maps. Based on requirements (Sec. 3.1), an overview (Sec. 3.2) and implementation details are described (Secs. 3.3 to 3.5).

3.1 Requirements and Preliminaries

According to the challenges related to existing visualization approaches using reachability maps (Sec. 1), requirements arise for a method to deal with these. A FCV technique for web-based reachability maps should therefore provide:

R1: an appropriate visualization to enable the user to view both, a detailed and a compact representation of reachability data within one display. This addresses challenges C1 and C3 to facilitate a more effective visual analysis of the geographic network. Accordingly, the visualization should provide users with a detailed view on relevant information without losing track of the context.

R2: an application responding on user interaction in real-time, addressing the need for interactivity of focus+context applications [28]. Since the user should be enabled to define a focal region by themselves, they will expect an instant visual feedback.

R3: an alternative method to server-based Geometry-based Isochrones (GBI) generation that reduces both, data transmission and processing effort. This addresses C2 and defines the need to develop an alternative method to the data-intensive conventional methods for isochrone generation. It is related to R2 as the isochrone generation method should be able to produce instant visual results according to modifications of user-defined parameters (e.g., the starting position or travel time ranges).

R4: a graphical user interface that assist users in the decision making process. This is important, as the proposed method does not primarily intend to predict ROI for the user, which is not the task of a reachability map. In fact, it aims to assist the user discovering a ROI. This can be achieved by allowing to arbitrarily define stylization parameters and other variables of the visual output. It is related to R2, since the real-time responses are essential for interactivity.

For these purposes, the Magic Lens metaphor stands out among the FCV techniques, as it enables users to interactively observe different visual appearances of the data while having full control over the ROI. It provides the most detailed representation of reachability information (i.e., the NBV) inside its outlines while keeping a generalized view (i.e., the IBV) surrounding. R2 and R3 will be addressed by applying sophisticated GPU-accelerated techniques on the client-side.

3.2 Conceptual Overview

Our approach extends the work of Schoedon *et al.* [23] and therefore builds on backend-provided glTF buffers, containing the street network and the respective travel times. After their initial rasterization, we are able to create the context representation out of the focus representation while making use of a sophisticated information distribution algorithm. The particular components of the FCV (i.e., focus, context, and ROI mask) will eventually be composed at the end of the rendering process. Fig. 2 shows a conceptual overview comprising the control and data flow between the following stages described in the remainder of this section:

Network-Rendering Stage: rasterizes the network geometry into a G-Buffer representation [22]. This step follows Schoedon et al. [23] and will therefore not be covered in this paper.

Isochrone-Generation Stage: comprises an initialization pass for generating an initial texture as input for subsequent flooding passes that distribute travel times over the map to obtain isochrones (Sec. 3.3).

Mask-Generation Stage: generates a mask texture representing the ROI (magic lenses or boundaries) and is triggered by user interaction (Sec. 3.4).

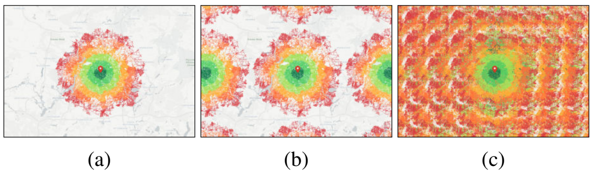


Figure 3: Visualization (including color mapping) of JFA steps for an input network (a): two rounds (b) and four rounds (c).

Color Mapping and Compositing Stage: uses the results of the network rendering pass, the flooding passes, and the mask-generation pass to create the respective mapping from travel time to color, and combines them into a single image (Sec. 3.5).

3.3 Image-based Isochrone Generation

Our method for the generation of Image-based Isochrones (IBI) is based upon distributing data from set pixels to adjacent non-attributed pixels, resulting in *Voronoi cells*. For it, we make use of the Jump-Flooding Algorithm (JFA) [21] that allows for fast propagation of pixels' contents to their closest neighbors in constant time whereas being independent of the number of input pixel elements (i.e., *seed points*) and capable of running in parallel on the GPU. Instead of applying CPU-consuming computations of isochrone geometry, we are therefore able to obtain the context representation directly from the focus representation by making use of GPU-accelerated image processing techniques.

JFA-driven Propagation of Travel Time. The JFA performs a flooding of information over a target space: given a grid of size $w \times h$ (i.e., the input image) containing the seed points which values should be propagated, the algorithm ensures that each grid cell (i.e., a pixel) has correspondence to its closest seed. For it, the JFA distributes a pixel's information in a total of $\log_2(\max(w,h))$ rounds with varying step lengths, using the information of already passed pixels. Its basic functionality is illustrated in Fig. 4, whereas Fig. 3 depicts different jump flooding steps for a colored input street network.

To obtain Voronoi diagrams, the flooded pixels ultimately need to acquire the desired information (typically, the color) from their corresponding seed points. Therefore, Rong and Tan suggest for using the spread seed's coordinates of the jump flooding result to refer to an initially created texture for picking up the respective color values and dying the final image accordingly. However, this procedure requires an additional rendering pass that applies an initial color mapping to the seed points and, moreover, complicates a real-time enabled and interactive stylization of the isochrones with custom color ramps (Sec. 3.5) since all jump flooding

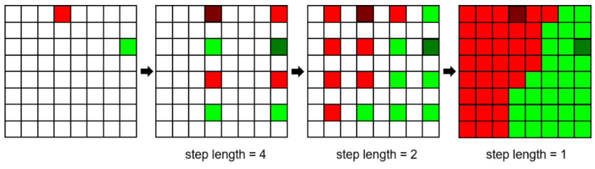


Figure 4: Working principle of JFA exemplified on an 8×8 grid and two seed points. Three rounds are required to fill the image.

passes need to be executed again whenever the style parameters are being changed. To avoid these issues, we adapt the original JFA approach by augmenting the information to be passed with another component. Alongside the seeds' positions we furthermore pass the travel time of the corresponding nodes of the transportation network by making use of an idle texture channel. In doing so, the color mapping can be carried out directly on the jump flooding result where every pixel of the viewport contains the travel time information of its closest supplying element and can be dyed according to that information.

WebGL Implementation Aspects. For the implementation of the JFA we make use of the WebGL Application Programming Interface (API) as it enables access to graphics hardware resources from a web browser using JavaScript (R2). Since our approach addresses image-based processing techniques and multiple rendering passes, the corresponding means of choice in WebGL are *textures*.

To ensure a proper functionality of the JFA, the following considerations about the textures' structure should be taken into account. WebGL 1.0 supports textures with four channels, i.e., Red-Green-Blue-α (RGBA). Since the individual channels store numeric values we can make use of them for image-based General-Purpose Computing on Graphics Processing Units (GPGPU) and store the x,y-coordinates of the seed points in the r,g-channels of the texture. The fragment shader's output variable in the OpenGL ES Shading Language (GLSL ES) 1.0 specification requires each element to range between 0.0 and 1.0. Thus, the seed's screen-space coordinates are normalized by dividing them by the resolution of the texture. Since our approach comprises furthermore the distribution of travel times, they are being stored in the alpha-component of the RGBA-texture. In addition, the blue-component will be used to store a Boolean-like value, indicating whether a pixel has already been visited by the flood or not.

By default, WebGL 1.0 assigns the `gl.UNSIGNED_BYTE` data type to texture objects which refers to 8 bit per channel and therefore 256 different values. The consequence is an internal mapping (clamp and quantization) of the $[0.0, 1.0]$ ranging floating point values to the $[0, 255]$ integral range which implies rounding and clamping operations. When retrieving the original

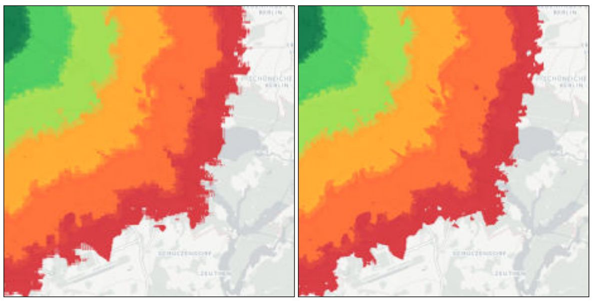

(a) 8-bit channel precision. (b) 32-bit channel precision.

Figure 5: Visual artifacts caused by precision issues, using a texture with unsigned byte precision (a) and with floating-point precision (b).

value through texture look-up, another internal remapping takes place, ensuring the texture values to be in $[0.0, 1.0]$ range again. Especially when dealing with high precise values, such as normalized screen-space coordinates of fragments, information is likely to get lost due to rounding issues, which could, in this case, result in visual artifacts caused by miscalculated seed positions, pointing to an incorrect pixel (Fig. 5). Therefore, we propose using a high-precision floating-point texture with a bit depth of 32 per channel, hence allowing for the storage of real numbers in the respective channels. This format is provided by the `OES_texture_float` extension and enables the definition of a `gl.FLOAT` texture, which prevents internal mappings of the normalized floating-point values.

Besides their main usage of determining a pixel's color on an object's surface based on images, textures can also be used as render targets in the form of G-Buffers [22] when they are attached as an additional color buffer to a custom Framebuffer Object (FBO). Since this technique enables rendering into a different target than the default framebuffer it is referred to as *off-screen rendering* which is a common tool in computer graphics for post-processing of previously rendered data to apply image-based effects. The geometry the off-screen texture is drawn to is considered a *screen-aligned quad*, since the application aims to display the processed texture on the whole canvas.

This concept of off-screen rendering and FBO-attached textures is indispensable for the implementation of the JFA, particularly because it is concerned an iterative multi-pass procedure where previously processed data needs to serve as input in the next iteration. To this effect, the principle of *ping-pong buffers* [2] allows for data processing between two off-screen textures, where the read-from-texture provides the input for the operation and the write-to-texture serves as render target for the processing results. After the execution of such a pass, or respectively, a JFA fragment shader, the two textures swap their roles. This procedure is repeated until a stop criterion has reached, in this case, a jump flooding step length of 1. The result of the JFA passes will be an off-screen texture, containing travel time values for every texel and accordingly being ready for color mapping (Sec. 3.5).

3.4 Mask Generation Stage

As mentioned in Sec. 1, we use the Magic Lens metaphor to combine the detailed NBV with the overview-giving IBV. In contrast to conventional lens approaches in information visualization [27], the lenses in the scope of our work do not literally execute a dedicated lens function, such as magnification or fish-eye distortion, that creates the focus representation on-the-fly.

Instead, our lens purely serves as a mask, represented by a function of its position and shape, to determine where the previously created textures will be displayed. To meet R2 and R4, we conceive intuitive and interactive lenses, since the ROI is hardly predictable. As a result, most of our lenses follow the circular shape of the historically well-known magnifying glass and are highly user-defined. The following lens types are supported:

Interactive Lens: This user-controlled lens is interactively moveable across the map according to the mouse cursor (Fig. 6a). Since the user makes a decision according to unknown preferences, this lens suits well for reachability analytics. Its position is therefore specified by the mouse pointer's screen coordinates and the radius can be user-defined.

Marker Lens: Since reachability analysis allows for the definition of multiple starting points, e.g., in the scope of business location analytics, the availability of multiple lenses could be meaningful too. Hence, automatic lens placement at respective markers is presented (Fig. 6b). They are placed around the center of a position marker and therefore highlight the immediate surroundings of a POI. They can have a static radius but also be defined according to user-specific parameters such as distance from a main location or several statistical variables.

Isochronal Lens: The shape of this lens type directly corresponds to the travel times while being positioned at respective markers (Fig. 6c). Given a specific travel-time range defined by the user, the NBV appears within this range while displaying isochrones in the surrounding context. According to the definition of [26], this can be denoted as a smart lens, as it considers data-driven aspects of picking out values within a specific range (travel times).

Depending on the user's choice of an appropriate lens type, the respective shape will be rendered into the off-screen mask texture and hence be available for real-time enabled blending operations in the compositing stage.

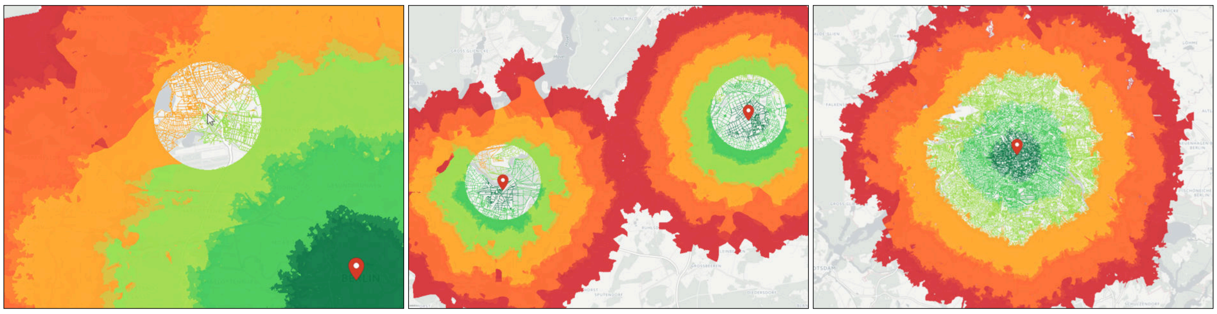

(a) Interactive Lens. (b) Marker Lens. (c) Isochronal Lens.

Figure 6: Different types of lenses supported by the presented approach.

3.5 Color Mapping and Compositing

The result of the jump flooding passes is not yet the desired IBV but merely an image with travel time values in the α-component of every pixel. To achieve an impression of areal isochrones, the pixels on the map have to be colored on the basis of their travel time. Therefore, a dedicated shader has to be executed to perform a sampling of a custom color ramp texture based on the normalized travel times of the jump flooding texture and make the resulting context texture available in the memory.

Since the JFA propagates information over the complete canvas (Fig. 3c), we furthermore required to request a higher maximum travel time from the routing server and accordingly clamp the travel times to be displayed with a lower value in the fragment shader by discarding fragments with a travel time value above the threshold. This also allows for a smooth user-defined adjustment of travel times, e.g., by using a slider.

Provided the mask texture, representing the lens's shape (Sec. 3.4), a final compositing pass combines the previously generated focus and context textures using *alpha blending* [2]. In GLSL ES, the *over* operator is natively implemented by the `mix` function that actually performs a linear interpolation between the color vectors of the focus and context textures using the mask's α-value. By setting the mask's α-values to 1.0 where the lens's shape should be displayed and to 0.0 elsewhere, the over operator merely performs a conditional function to determine if either the focus texture or the context texture should be rendered.

Further variations of the lens presentation can be achieved by using a gradient mask texture with varying α-values for the lens's fragments [24]. By decreasing the α-value from the lens interior outwards, a smoother transition between focus regions and the context can be accomplished. This technique therefore represents a more strictly adaption to the focus+context definition of Cockburn [6] who stresses a seamless transition between focus and context to be required for FCVs.

4 RESULTS AND DISCUSSION

This section discusses the results of the proposed approach by evaluating our method for the generation of isochrones (Sec. 4.1), followed by a performance measurement using different web-browsers on mobile clients (Sec. 4.2), and a discussion of limitations as well as future research directions (Sec. 4.3).

4.1 Isochrone Evaluation

One main contribution of this work is the development of an alternative, image-based technique for generating isochrones of travel time data. It is hence worthwhile to examine the results of our approach and to compare them with established isochrone generation methods.

Accuracy Issues. Fig. 7 illustrates a noteworthy behavior of our IBI approach, associated with the JFA. Referring to [23], we allow for the selection of different transportation media (e.g., car, bike, public transit) in our web application. The JFA, however, will propagate also travel time information from street segments that are not accessible by the specified medium – a motorway, e.g., is forbidden for cyclists – and therefore causes remarkable gaps inside the isochrones due to subsequent discarding of fragments (Sec. 3.5). Although this behavior depicts a correct representation of the underlying data, it provokes a considerable amount of visual clutter and makes it difficult to distinguish any spatial relations in the disrupted isochrones (Fig. 7a). Since the task of the context in a FCV, though, is to provide an overview of the data, we suggest to carry out a pre-filtering of the travel time data by assigning those gap-causing fragments a specific value in the initialization pass, indicating they are not considered seed points and therefore not being distributed by the JFA. The resulting loss of accuracy, however, can be compensated by using our proposed Magic Lens as the detail information of the gaps will be shifted to the focus representation inside the lens.

Comparison of Approaches. A comparison of visualization results using IBI, as proposed in this paper, and the conventional approach of server-side generated GBI

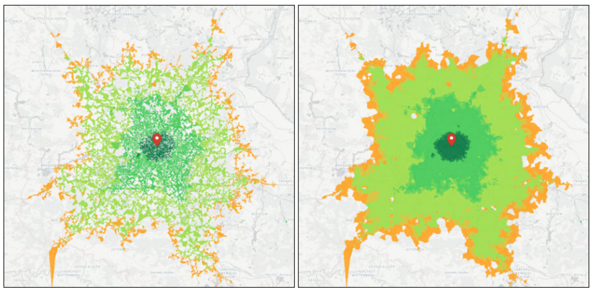

(a) Unfiltered travel time data, causing large amount of gaps. (b) Pre-filtering of travel time for flooding into invalid cells.

Figure 7: Comparison of results using unfiltered and pre-filtered travel time data.

is depicted in Fig. 8. Since the underlying algorithms of both approaches are completely different, a comparison can rather happen on the basis of visual results. Therefore, a single-marker application for both approaches is established that displays isochrones from 5 to 30 minutes of travel time in five-minutes steps. Although they appear rather similar, the GBI provides a smoother appearance than the IBI, which provides sharp-edged structures due to the nature of the JFA. Furthermore, it can be noticed that the IBI approach offers a considerable higher Level-of-Detail (LOD) whereas the GBI approach performs a server-side generalization that may result in inconsistencies regarding a correct representation of travel times, as some areas are depicted reachable though they are actually not.

However, the LOD of the GBI varies with different zoom levels, which can be considered a desired behavior. The IBI approach is yet not able to vary its LOD, though the aforementioned method of filling the gaps is an attempt to simulate a lower LOD. Due to the need for zooming into the map to survey detailed representations with the GBI approach, the user might lose track of the overall situation, whereas the IBI in combination with the Magic Lens allow for a combined visualization of detail and overview within one viewport. For that reason and since both approaches have a similar appearance whereas the IBI provides more detail, it can be considered an adequate alternative to the established GBI approaches.

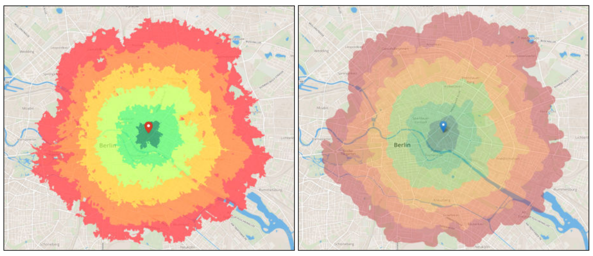

(a) Image-based isochones. (b) Geometry-based isochones.

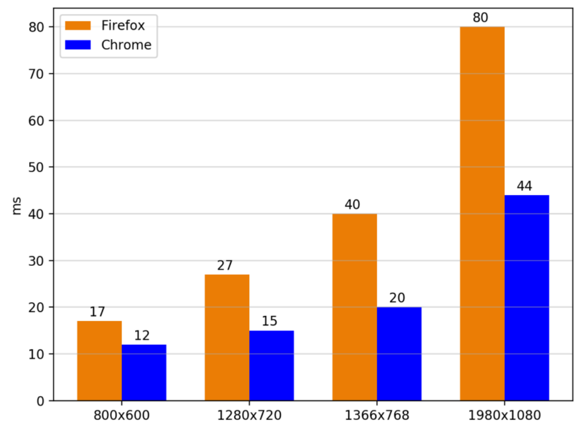
Figure 8: Comparison of the isochrones computed (a) by the presented client-side IBI and (b) by the existing server-side GBI.

4.2 Rendering Performance

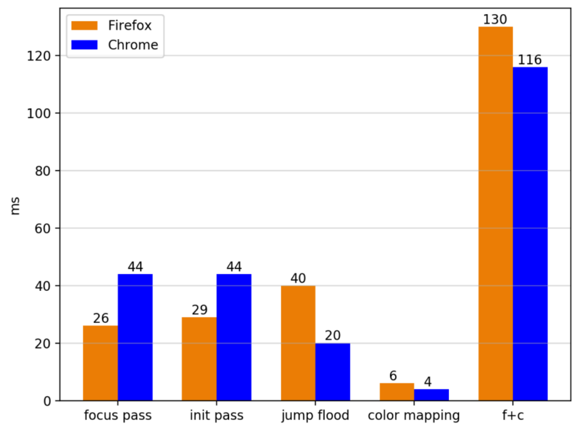
Test Data and Hardware. The data to be rendered is a tiled glTF response [23] containing 659 186 vertices and 329 593 line primitives. All tiles are loaded into a tile cache before executing the rendering operations. The WebGL rendering performance measurements are performed on a Microsoft Surface Pro 4 equipped with an Intel Core i7-6650U 2.2 GHz Processor and 16 GB RAM at 1867 MHz. The GPU is an integrated Intel Iris Graphics 540 with 128 MB dedicated VRAM and 8154 MB shared system memory. The respective measurements are performed in full-screen mode on the Surface Display with the half of the native resolution of 2736×1824 pixels at 267 ppi. The test application operates on Microsoft Windows 10 Pro operating system with a 64 bit architecture. The utilized browsers are Mozilla Firefox (Version 55.0.3, Mozilla renderer) and Google Chrome (Version 60.0.3112.113, Webkit renderer), both with WebGL 1.0/GLSL ES 1.0 support.

Test Procedure. This work focuses on the frontend rendering, which is why the performance evaluation is limited to the measurement of WebGL draw calls for the JFA and the individual rendering passes. Due to the postulated independence of the JFA on the number of input seeds [21], we compare the performance of the jump flooding passes among different screen sizes, as the JFA is stated to be dependent on the texture size. To obtain meaningful results, the JFA draw call is wrapped in a loop, executing 500 times and accordingly being summed and averaged. Since the JFA is a round-based algorithm that updates its input (i.e., textures and step size) in every round, these values have to be reset after every sample measurement to ensure the JFA to operate properly by using the same input every time. Fig. 9a shows the plots of the measurements of the JFA for the two respective browsers.

Test Results. Despite Google Chrome performs constantly better than Mozilla Firefox, the dependency of the speed on the screen size becomes obvious. However, since the isochrone generation is likely to have no need for real-time responses as they are computed only on user-specific map interaction (setting of markers, zooming), this behavior is not of outstanding importance. Instead, the use of caching logic allows for an increasing speed of the JFA, since there is less space for the seed points to flood into. Fig. 9b compares the rendering speed of the prominent rendering passes; the left plot accumulates the other four ones. A behavior worth mentioning is the inferior performance of Google Chrome compared to Firefox in the focus and initialization passes. This seems to be provoked by CPU-intense operations such as matrix transformations or leaflet-related procedures. In contrast thereto, in the JFA and color mapping passes, Chrome outperforms Firefox, since there are no matrix computations to be performed.



(a) Jump flooding rendering performance.



(b) Comparison of rendering passes.

Figure 9: Measurement results of rendering performances with the browsers Mozilla Firefox and Google Chrome. Measurements are performed for (a) the jump flooding algorithm and (b) the rendering passes executed to create Focus and Context views. The performance is measured in milliseconds (ms).

4.3 Limitations and Future Research

Conceptual and Technical Limitations. The proposed technique aims to enable users to effectively extract relevant information from a reachability map. However, due to a considerable amount of pixels representing street segments in the focus area at low zoom levels, one can hardly distinguish individual streets, particularly where the network density is high. The strengths of the NBV [23] thus become apparent primarily at high zoom levels, whereas IBV are sufficient for low zoom level applications. Remedy may be providing a combination with a magnification lens or rather a fish-eye-like projection [20].

The interactive mouse lens (Fig. 6a) allows for a user-defined determination of a ROI, while the isochronal lens (Fig. 6c) attempts to anticipate the relevance of the data by assuming it to be a function of reachability. However, the general purpose of a reachability map is to support the user in yet determining a ROI. Since the approach cannot predict the user's specific interest, the isochronal lens as well as the marker lens (Fig. 6b) are biased in some respects, as a region or object of the user's interest might possibly be located in areas difficult to reach.

Another aspect to consider when dealing with isochrones is the cartographic generalization implicitly applied due to the definition of Voronoi diagrams. As a result, travel time values may be assigned to parts of the map where inherently no information is available and may therefore cause interpretation errors. Technical limitations are furthermore introduced according to the nature of the JFA and the tiling principle, which causes streaks during the asynchronous image composition stage. Besides, the WebGL technology and its extensions lack cross-browser availability.

Future Research Directions. To prove the acceptability of the proposed approach, further user studies have to be conducted. Though the presented work focuses on reachability analytics and a respective visualization using reachability maps, the proposed technique of creating IBI using JFA is not limited to this subject, as it is able to propagate arbitrary information derived from different geographic feature representations beyond networks and can be used in a wider range of applications. Furthermore, another post-processing steps could be envisioned to obtain smoother shapes for the isochrones.

5 CONCLUSIONS

This paper introduces a novel approach to apply lens-based FCV techniques to interactive web-based reachability maps. It presents three types of lenses to provide the user with both, a detailed and a contextual representation of reachability information within a single view and supports the user to extract detailed reachability information while retaining contextual overview. To attain an interactive and responsive application, the web-based implementation takes advantage of graphics hardware using the WebGL standard for GPU programming. Comparisons with geometry-based isochrone generation methods indicate the exchangeable applicability of the presented image-based isochrones.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the paper. This work was funded by the Federal Ministry of Education and Research (BMBF), Germany within the “MOBIE” project (www.mobie-project.de).

REFERENCES

[1] Maneesh Agrawala and Chris Stolte. Rendering effective route maps: Improving usability through generalization. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 241–249, New York, NY, USA, 2001. ACM.

[2] Tomas Akenine-Möller, Eric Haines, Naty Hoffman, Angelo Pesce, Michał Iwanicki, and Sébastien Hillaire. *Real-Time Rendering 4th Edition*. A K Peters/CRC Press, Boca Raton, FL, USA, 2018.

[3] Harvey W. Armstrong. A network analysis of airport accessibility in south hampshire. *Journal of Transport Economics and Policy*, pages 294–307, 1972.

[4] Staffan Björk, Lars Erik Holmquist, and Johan Redström. A framework for focus+context visualization. In *Proceedings of the 1999 IEEE Symposium on Information Visualization*, INFOVIS '99, pages 53–, Washington, DC, USA, 1999. IEEE Computer Society.

[5] Tom Carden. Travel time tube map, 2006. Accessed: 2019-04-40.

[6] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Comput. Surv.*, 41(1):2:1–2:31, January 2009.

[7] Georg Fuchs, Matthias Kreuseler, and Heidrun Schumann. Extended focus & context for visualizing abstract data on maps. In *CODATA Prague Workshop - Information Visualization, Presentation, and Design*, 03 2004.

[8] G. W. Furnas. Generalized fisheye views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '86, pages 16–23, New York, NY, USA, 1986. ACM.

[9] Francis Galton. On the construction of isochronic passage-charts. In *Proceedings of the Royal Geographical Society and Monthly Record of Geography*, volume 3, pages 657–658. JSTOR, 1881.

[10] Jan-Henrik Haunert and Leon Sering. Drawing road networks with focus regions. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2555–2562, dec 2011.

[11] Helwig Hauser. *Scientific Visualization: The Visual Extraction of Knowledge from Data*, chapter Generalizing Focus+Context Visualization, pages 305–327. Springer Berlin Heidelberg, 2006.

[12] Henning Hollburg, Christoph Sinn, and Patrick Voland. *Angewandte Geoinformatik 2012.*, chapter Hier bin ich - was kann ich erreichen? Webbasierte, interaktive Erreichbarkeitsanalyse touristischer Ziele der Stadt Potsdam, pages 317–322. Herbert Wichmann Verlag, VDE VERLAG GMBH, Berlin/Offenbach, 2012.

[13] Ishantha Lokuge; Suguru Ishizaki. Geospace: An interactive visualization system for exploring complex information spaces. *Proceedings of the ACM CHI '95 Conference on Human Factors in Computing Systems*, pages 409–414, 1995.

[14] P. Karnick, D. Cline, S. Jeschke, A. Razdan, and P. Wonka. Route visualization using detail lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):235–247, March 2010.

[15] Johannes Kopf, Maneesh Agrawala, David Bargeron, David Salesin, and Michael Cohen. Automatic generation of destination maps. *ACM Trans. Graph.*, 29(6):158:1–158:12, December 2010.

[16] Robert Kosara, Helwig Hauser, and Donna L. Gresh. An Interaction View on Information Visualization. In *Eurographics 2003 - STARs*. Eurographics Association, 2003.

[17] Josua Krause, Marc Spicker, Leonard Wörteler, Matthias Schäfer, Leishi Zhang, and Hendrik Strobelt. Interactive visualization for real-time public transport journey planning. In *Proceedings of SIGRAD 2012*, number 81, pages 95–98. Linköping University Electronic Press; Linköpings universitet, 2012.

[18] Meinhard Lemke, Carsten Schürmann, Klaus Spiekermann, and Michael Wegener. *Nationalatlas Bundesrepublik Deutschland - Verkehr und Kommunikation*, chapter Transeuropäische Verkehrsnetze, pages 42–45. Institut f. Länderkunde, 2001.

[19] Vincent Meertens. Featured graphic. travel time maps of urban areas in the netherlands. In *Environment and Planning A - Volume 44*, 2012.

[20] Emmanuel Pietriga, Olivier Bau, and Caroline Appert. Representation-independent in-place magnification with sigma lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):455–467, May 2010.

[21] Guodong Rong and Tiow-Seng Tan. Jump flooding in gpu with applications to voronoi diagrams and distance transform. In *I3D '06 Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, 03 2006.

[22] Takafumi Saito and Tokiichiro Takahashi. Comprehensible rendering of 3-d shapes. *SIGGRAPH Comput. Graph.*, 24(4):197–206, September 1990.

[23] Alexander Schoedon, Matthias Trapp, Henning Hollburg, and Jürgen Döllner. Interactive Web-based Visualization for Accessibility Mapping of Transportation Networks. In *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers*, EuroVis '16, pages 79–83, Goslar Germany, Germany, 2016. Eurographics Association.

[24] Amir Semmo, Matthias Trapp, Jan Eric Kyprianidis, and Jürgen Döllner. Interactive visualization of generalized virtual 3d city models using level-of-abstraction transitions. *Comput. Graph. Forum*, 31(3pt1):885–894, June 2012.

[25] Klaus Spiekermann. Visualisierung von eisenbahnreisezeiten - ein interaktives computerprogramm. Final project report, Institut für Raumplanung, Universität Dortmund, 1999. german.

[26] Conrad Thiede, Georg Fuchs, and Heidrun Schumann. *Smart Lenses*, pages 178–189. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[27] Christian Tominski, Stefan Gladisch, Ulrike Kister, Raimund Dachselt, and Heidrun Schumann. A Survey on Interactive Lenses in Visualization. In R. Borgo, R. Maciejewski, and I. Viola, editors, *EuroVis - STARs*. The Eurographics Association, 2014.

[28] Matthias Trapp. *Interactive Rendering Techniques for Focus+Context Visualization of 3D Geovirtual Environments*. PhD thesis, Hasso-Plattner-Institut, Mathematisch-Naturwissenschaftliche Fakultät, Universität Potsdam, 2013.

[29] Matthias Trapp, Christian Beesk, Sebastian Pasewaldt, and Jürgen Döllner. Interactive Rendering Techniques for Highlighting in 3D Geovirtual Environments. In *Advances in 3D Geo-Information Sciences*, volume 9 of *Lecture Notes in Geoinformation and Cartography*, pages 197–210. Springer Berlin Heidelberg, January 2011.

[30] Matthias Trapp, Amir Semmo, and Jürgen Döllner. Interactive rendering and stylization of transportation networks using distance fields. In *Proceedings of the 10th International Conference on Computer Graphics Theory and Applications*, GRAPP 2015, pages 207–219, Portugal, 2015. SCITEPRESS.

[31] Yu-Shen Wang and Ming-Te Chi. Focus+context metro maps. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2528–2535, Dec 2011.

[32] Shi Yin, Moyin Li, Nebiyou Tilahun, Angus Forbes, and Andrew Johnson. Understanding transportation accessibility of metropolitan chicago through interactive visualization. In *Proceedings of the 1st International ACM SIGSPATIAL Workshop on Smart Cities and Urban Analytics*, UrbanGIS'15, pages 77–84, New York, NY, USA, 2015. ACM.